A 2.5 V CMOS Delay-Locked Loop for an 18 Mbit, 500 Megabyte/s DRAM

Thomas H. Lee, Kevin S. Donnelly, John T. C. Ho, Jared Zerbe, Mark G. Johnson, and Toru Ishikawa

Abstract—**This paper describes clock recovery circuits specifically designed for the hostile noise environment found aboard dynamic RAM chips. Instead of a phase-locked loop having a voltage-controlled oscillator, these circuits implement a delay-locked loop, thereby achieving low jitter and reduced sensitivity to noise on the substrate and the power supply rails. Differential signals are employed both in signal paths and in control paths, further decreasing noise sensitivity and simultaneously allowing operation from low voltage supplies. An unorthodox voltage-controlled phase shifter, operating on the principle of quadrature mixing, yields a circuit with unlimited delay range (modulo 2π radians). Minor loops, enclosed within the overall loop feedback path, perform active duty cycle correction. Measured results show peak-to-peak jitter of 140 ps on the internal clock signal, and 250 ps on the external data pins, sufficiently small to allow 500 Megabyte/s transfer rates at the I/O interface.**

I. Introduction

A NARROW, high-speed bus provides large bandwidth in a small, low pin-count package, but places great demands on timing accuracy. While VCO-based phase-locked loops (PLL's) have been used successfully to provide compensation for a variety of on-chip sources of timing errors in 5 V DRAM's [1], it becomes increasingly difficult to obtain acceptable power supply noise-induced jitter performance from such a PLL as supply voltages drop. Additionally, acquisition time is often longer than desirable (e.g., several microseconds), due to the time it takes to drive the VCO to the correct frequency.

While such difficulties might be surmountable, it is worthwhile to observe that there is no fundamental need for a VCO if the system already provides a clock of the correct frequency; the problem is simply to place the edges of this clock precisely. This paper thus describes a pair of *delay-locked loops* (one DLL for transmitting data, one for receiving) that satisfy the requirement for accurate timing (sub-100 ps static phase error), even in the noisy substrate and VDD environment of DRAM's, to allow data transfer at rates exceeding 500 Mbits/s/pin at 2.5 V.

While the application of delay-locked loops to the problem of host-slave synchronization [2] is certainly not new, the loop described in this paper solves several problems of conventional PLL's and DLL's, providing unlimited phase shift (modulo 2π) without a VCO, enabling lock in under 200 ns, and exhibiting good jitter performance at low supply voltages.

Manuscript recieved May 13, 1994; revised August 21, 1994.

T. H. Lee, K. Donnelly, J. Ho, J. Zerbe, and M. Johnson are with Rambus Inc., Mountain View, CA 94040-1419.

T. Ishikawa is with the LSI Memory Division, NEC Corporation, Sagamihara, Kanagawa 229, Japan.

IEEE Log Number 9406273.

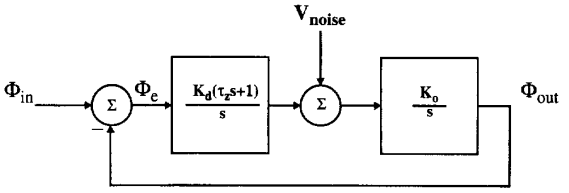

Fig. 1. Linearized PLL noise model.

II. Review of PLL Noise Performance

To motivate the present work, let us examine the behavior of a conventional, second-order, VCO-based PLL in the presence of noise. The linear model of Fig. 1 considers noise as additive at the control port of the VCO. The phase detector gain is K_D volts/radian, the VCO gain constant is K_O radians/V-s, and τ_z is the time constant of the loop-stabilizing zero.

It is a relatively straightforward exercise to show that the noise-to-phase-error transfer function for this system is:

$$-\frac{\phi_\epsilon}{V_N} = \frac{sK_O}{s^2 + s\tau_z K_D K_O + K_D K_O} \tag{2.1}$$

Assuming a unit step function noise input, the resulting phase error as a function of time may be expressed as:

$$-\phi_\epsilon(t) = \frac{\Delta\omega_i}{\omega_n\sqrt{\zeta^2-1}}\exp(-\zeta\omega_n t)\sinh\left(\omega_n\sqrt{\zeta^2-1}t\right) \tag{2.2}$$

where $\Delta\omega_i$ is the initial frequency error due to the step-function disturbance V_N. The damping ratio of the closed-loop poles, ζ, is given by:

$$\zeta = \frac{\omega_n\tau_z}{2} \tag{2.3}$$

and ω_n, the natural frequency, by:

$$\omega_n = \sqrt{K_D K_O} \tag{2.4}$$

The maximum phase error is found from (2.2) to be

$$-\phi_{\epsilon,\max} = \frac{\Delta\omega_i}{\omega_n\sqrt{\zeta^2-1}}\exp\left(-\frac{\zeta}{\sqrt{\zeta^2-1}}\operatorname{atanh}\frac{\sqrt{\zeta^2-1}}{\zeta}\right) \times \sinh\left(\operatorname{atanh}\frac{\sqrt{\zeta^2-1}}{\zeta}\right) \tag{2.5}$$

and occurs at a time

$$t_{\max} = \frac{1}{\omega_n\sqrt{\zeta^2-1}}\operatorname{atanh}\frac{\sqrt{\zeta^2-1}}{\zeta} \tag{2.6}$$

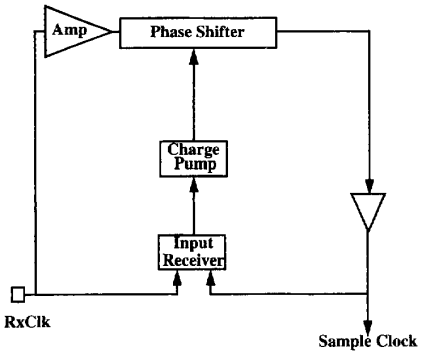

Fig. 2. Simplified RDLL block diagram.

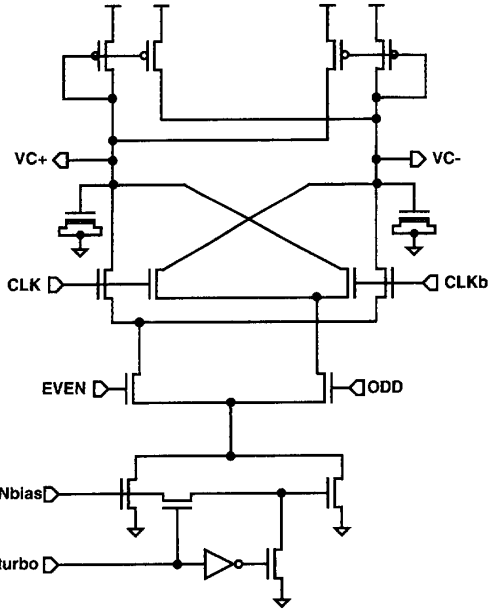

Fig. 3. Main charge pump.

Now, while the utility of the foregoing equations may be somewhat elusive owing to their cumbersome nature, these expressions simplify considerably in the limit of high damping:

$$-\phi_{\epsilon,\max} \approx \frac{\Delta\omega_i}{\omega_c} \tag{2.7}$$

$$t_{\max} \approx \frac{2\ln 2\zeta}{\omega_c} \tag{2.8}$$

where ω_c is the crossover frequency of the loop.

We see that the maximum phase error is approximately the ratio of the initial VCO frequency shift to the loop crossover frequency (which is approximately the closed-loop bandwidth). This relationship may be understood intuitively by recalling that phase is the integral of frequency, so that any departure from the correct frequency causes some integration of phase error, and that this accumulation of phase error persists for a time on the order of the reciprocal loop bandwidth.

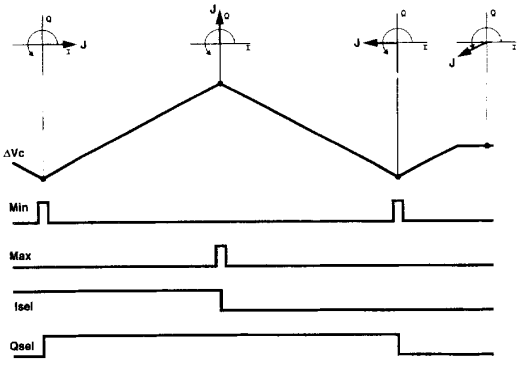

Fig. 4. Operation of phase shifter.

Minimizing jitter due to power supply noise therefore requires maximizing loop bandwidth and minimizing the initial shift in the VCO frequency. Unfortunately, arbitrarily large loop bandwidths are not possible because all practical feedback systems ultimately suffer phase margin degradation from a variety of sources, such as (possibly) poorly modeled parasitic elements. Additionally, many PLL's are sampled-data systems (phase error measurements are made at discrete intervals), and this nature imposes further bounds on the crossover frequency if stability is not to be compromised. Typically these considerations, in conjunction with the need to absorb component tolerances and drifts with temperature and supply voltage, force the use of loop bandwidths that are only a small fraction (e.g., <10%) of the clock frequency to guarantee acceptable worst-case phase margins.

At this point, a numerical example may be useful in underscoring the magnitude of the problem. Suppose that, for some value of step disturbance on the supplies, $\Delta\omega_i$ is 2% of ω_{clock}. Further assume that the crossover frequency, ω_c, of the loop is also 2% of ω_{clock}. In this case, the maximum phase error is one radian, or about 630 ps, with a clock a period of 4 ns. This magnitude of jitter is generally intolerable.

This jitter performance may be compared with that of a simple delay-locked loop. In this case there is no VCO and, hence, no mechanism for the accumulation of phase error due to supply noise. The cognate phase error equation for this type of DLL is therefore simply:

$$\phi_{\epsilon,\max} = \frac{2\pi\Delta T_D}{T_{\text{clock}}} \tag{2.9}$$

where T_D is the time delay of the phase-shifting element, and T_{clock} is the period of the clock.

Assuming that T_D has a value equal to the period of 4 ns, a 2% change in this delay corresponds to an 80 ps error, roughly a factor of 8 smaller than the VCO-based PLL. While this calculation, performed with artificial but representative numbers, should not be taken as proof of the superiority of DLL's in the presence of noise, it does suggest their utility in the present context.

III. Description of the DLL

A simplified block diagram for the receive DLL is shown in Fig. 2. The job of the receive DLL (RDLL) is to generate a

clock for optimum sampling of the input data, independent of process, temperature and data receiver setup time variations. In all that follows, assume that input data transitions are centered between the transitions of the incoming clock.

If the data receivers that sample the incoming data possessed no set-up time, the ideal local sampling clock would be perfectly in phase with the incoming clock, and the need for a loop would diminish. However, since all practical data receivers have some nonzero (and generally poorly-controlled) set-up time, the optimum sampling instant is displaced from transitions on the incoming clock by some amount. To produce a sampling clock delayed by precisely this displacement, *the same data receiver as is used in the I/O pad* is used as the RDLL's phase detector. The data receiver samples the incoming clock with the sampling clock generated by the RDLL, and the loop continually servos a phase shifter so that the output of the data receiver is high as often as it is low, thus producing a sample clock that is timed optimally, independent of variations in temperature, supply and process.

To produce the control voltage for the phase shifter, a differential charge pump is used to integrate the output signal of the phase detecting data receiver. A *negative differential resistance* formed by the cross-connected PMOS devices shown in Fig. 3 is placed in parallel with the diode-connected MOS loads to yield a net differential resistance that is (ideally) infinite to improve the quality of the integration. The *common-mode* problem of other approaches (e.g., current sources as loads) is eliminated by the low *common-mode* resistance of this connection, and thus no feedback is required to set the *common-mode* level (here, a gate-to-source voltage below VDD), enabling fast acquisition. As seen in the figure, this charge pump has an additional differential pair and pump direction control inputs ("even" and "odd") to allow the pump polarity to reverse at each quadrant crossing, necessary to allow a finite control voltage range to map into a continuous and unlimited phase shift range.

Operation of the phase shifter is shown in Fig. 4. To evade the finite phase shift range problems of prior-art delay-locked loops, this DLL uses *quadrature mixing* to provide a continuous and unlimited phase shift range. Furthermore, and in contrast with most DLL implementations, this particular phase shifter allows for a completely differential path for the control signals, further aiding power-supply rejection.

In tandem with the reversible charge pump, this DLL employs seamless switching (controlled by the phase selector finite-state machine) of the signals fed to the phase shifter. Referring to Fig. 4, consider the control voltage (ΔVC) while J varies from being in phase with I, to being in phase with I_b. Assume that ΔVC is at a minimum when J is in phase with I. As ΔVC increases, J approaches Q in phase, until ΔVC is at a maximum. When J is exactly aligned with Q, I may be replaced by I_b without causing any jitter (because the weight on I is zero). The control voltage now decreases from its maximum, causing a greater contribution of I_b and progressively less of Q. Eventually ΔVC returns to a minimum value, and J is aligned with I_b. Thus, any phase shift can be generated without running out of dynamic range on ΔVC; a finite ΔVC range allows the spanning of the entire

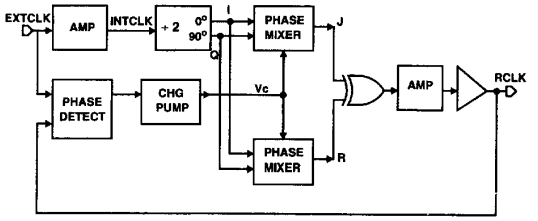

Fig. 5. More complete RDLL block diagram.

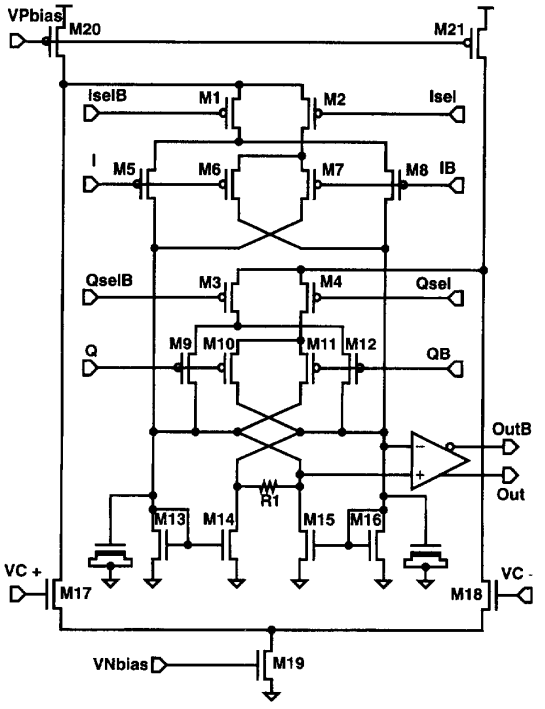

Fig. 6. Phase mixer.

phase space. Note also that while monotonicity is important, linearity is not, because the phase shifter is enclosed within a feedback loop.

As seen in Fig. 5, the input signals for the phase shifter are ultimately derived from a clock amplifier that converts the low-swing, quasi-differential bus clock signal to a CMOS-level signal. The amplified clock signal drives a quadrature divider, whose outputs are at half the frequency of the input and 90° apart. The quadrature outputs (and their complements) then feed a *phase mixer* where they are combined in a voltage-controlled ratio.

The schematic of the phase mixer is shown in Fig. 6. Note that, despite the vertical stacks of transistors, this circuit is compatible with operation at low supply voltages. Transistors $M1$–$M4$ simply act as switches under the control of the phase selector, $M5$–$M12$ are switches driven by the input (quadrature) clocks, while the differential control voltage from the charge pump feeds the differential pair $M17$–$M18$. A load cell ($M13$–$M16$) similar to that used in the charge

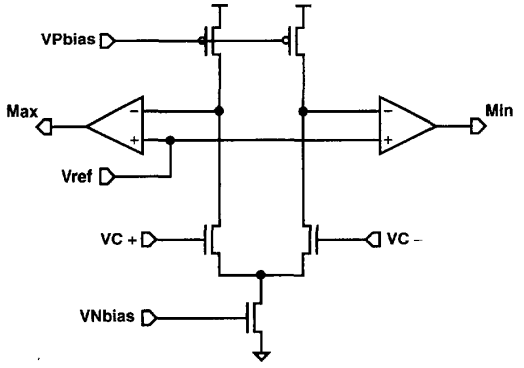

Fig. 7. Quadrant boundary detector.

pump expediently sets the common-mode level, while resistor $R1$ controls the asymptotic differential swing. The differential control voltage determines the charging current into the capacitors and therefore the weight of the selected, half-frequency phases. Because the resultant phase-shifted output is at half frequency, two phase mixers are used whose inputs are themselves in quadrature, and their outputs (J and R) XOR'd to regenerate a full-frequency signal.

To prevent jitter at quadrant crossings, it is important to permit the replacement of a vector with its complement only when its weight truly is zero. The quadrant boundary detector shown in Fig. 7 employs replicas of the differential pair and current sources of the phase mixers. The outputs ("max" and "min") of the quadrant boundary detector signal the *phase selector* finite-state machine (FSM) when the differential input voltage is so large that the weight of a phase has decreased to near enough zero that it can be replaced seamlessly with its complement.

Because a DLL derives its output signal from the input signal, it is sensitive to the duty cycle of the incoming clock. In the present work, it is assumed that data are transferred on both the rising and falling edges of the clock, so any departure from a 50% duty cycle degrades timing margins. To solve this problem, this DLL employs active duty cycle correction (DCC). The bus clock input signal whose duty cycle is to be corrected drives a DCC charge pump (similar to that used to generate the main control voltage) as shown in Fig. 8. If the clock signal has any duty cycle distortion, a net charge per cycle is deposited on the charge pump capacitors, causing the differential output voltage to ramp linearly; the output of the charge pump is thus proportional to the time integral of the duty cycle error. This output then feeds an extra port of the clock amplifier and effectively adds a variable offset to the input signal of the amplifier to adjust the duty cycle, as seen in Fig. 9.

Fig. 10 is a detailed block diagram of the RDLL. As can be seen, the complete loop includes a second duty cycle corrector to compensate for any distortion caused by the phase mixer and clock buffer chain.

In this type of loop, the control signal is not proportional to the magnitude of the phase error because the phase detector provides only a binary phase error indication. Hence, in

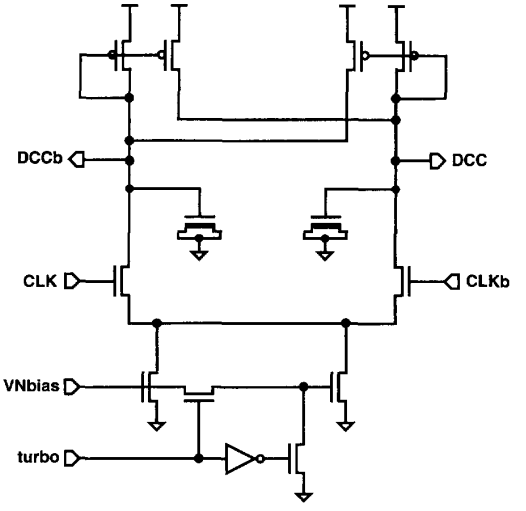

Fig. 8. DCC charge pump.

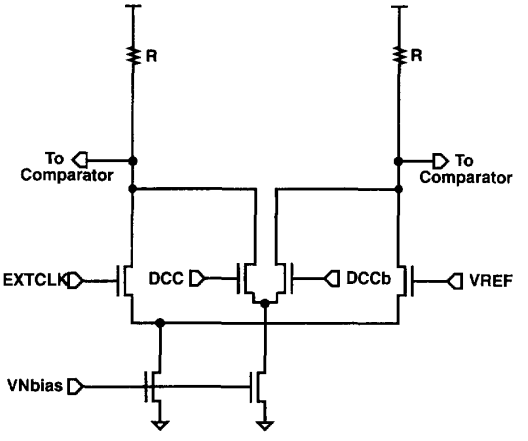

Fig. 9. Simplified clock amplifier with DCC port.

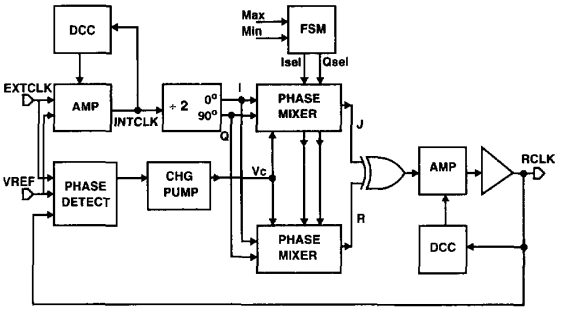

Fig. 10. Detailed block of RDLL diagram.

lock, this "bang-bang" loop dithers with some amplitude about the desired phase value. While a large charge pump current-to-capacitance ratio is desirable for fast acquisition, the corresponding dither amplitude generally would be excessive. To obtain fast acquisition without excessive dither

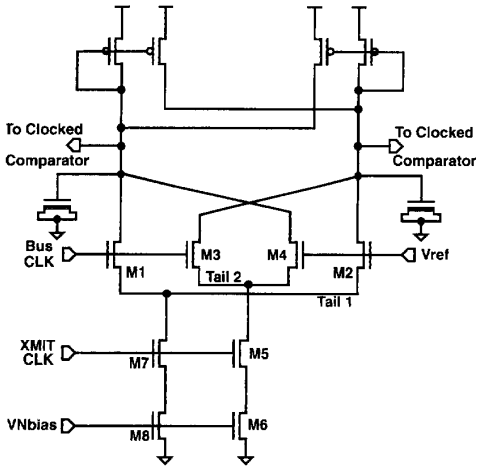

Fig. 11. Simplified quadrature phase detector.

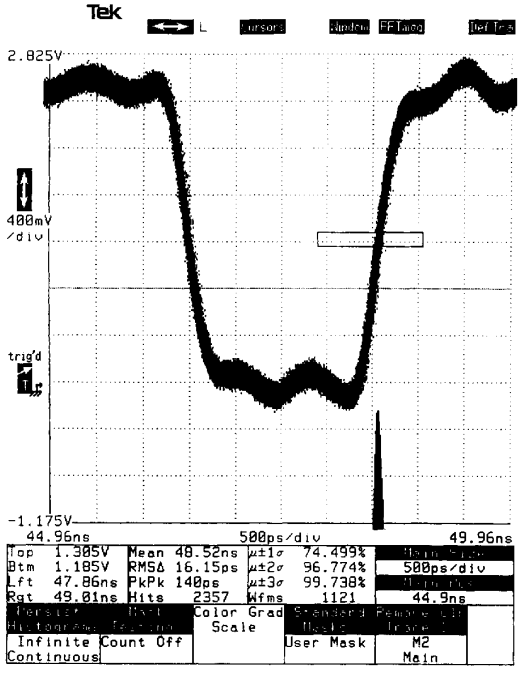

Fig. 12. Jitter histogram of RDLL @ 2.5 V, 250 MHz.

jitter, all charge pump currents are initially "turbo" boosted for approximately 250 ns, then reduced to minimize loop dither jitter.

The transmit DLL (TDLL) is similar is similar to the RDLL, except that the quadrature phase detector (QPD) shown in Fig. 11 is used to provide a transmit clock that is phase-shifted by 90° (recall that data transitions are centered between clock transitions). As seen in the figure, the QPD integrates the bus clock's voltage over a time window centered about the clock transition. Any departure from the desired quadrature condition leaves a nonzero net value at the end of the integration period. The core of the QPD then drives a clocked comparator that

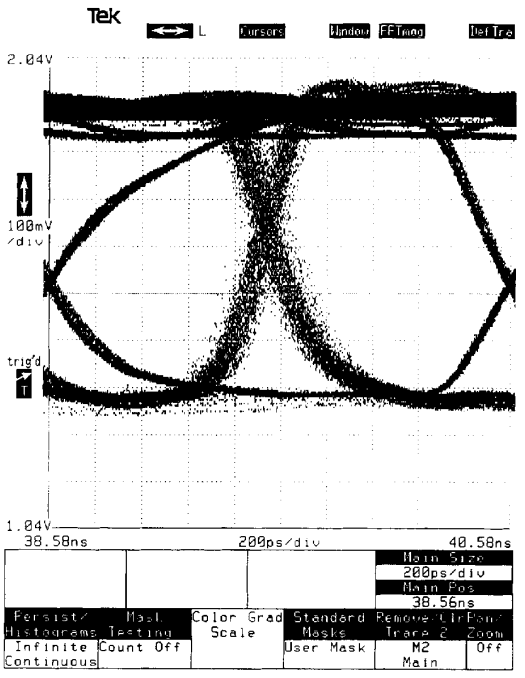

Fig. 13. Eye diagram at data I/O pin @ 500 MB/s.

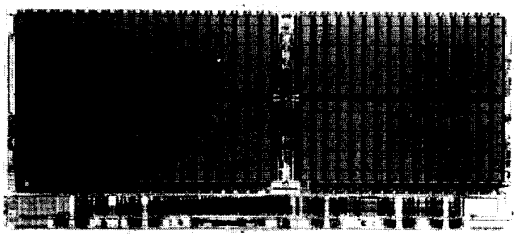

Fig. 14. Photomicrograph of DLL's.

provides a digital "early/late" signal analogous to that of the phase detector in the RDLL. The balance of the transmit loop is identical with the receive loop.

As mentioned previously, the input signals are quasi-differential, that is, the inputs are a dc reference voltage ("Vref") and a single-ended signal ("Bus Clk") with its small swing centered about this dc value. Because the input signal to the QPD thus possesses a common-mode component of high frequency, charge injection resulting from the presence of any capacitance at the node marked "Tail 1" causes an integration error with a corresponding phase offset that is typically unacceptably high (e.g., approximately 500 ps). To reduce this phase offset, this QPD employs a charge cancellation scheme (comprising transistors $M3$–$M6$). Transistors $M1$–$M4$ are identical, but $M6$ is made smaller than $M8$. Because the capacitances at both tail nodes are the same, the cancellation circuitry thus provides a charge injection that is equal in magnitude to, but opposite in polarity from, that induced in the core of the QPD by the common-mode variation in the input. With this modification, static phase offsets of under 100 ps are achieved.

IV. Experimental Results

The receive and transmit DLL's, along with the rest of a high- speed 18 Mbit DRAM, were fabricated in an unmodified 0.6 μm, double-metal CMOS DRAM process.

Fig. 12 is a jitter histogram of the internal receive (sampling) clock with no activity in the DRAM core. As can be seen, the peak-to-peak jitter is 140 ps at 250 MHz with a 2.5 V supply. Simulations also show an overall jitter sensitivity of approximately 1 ps/mV to disturbances on both supplies.

Perhaps more significant is the eye diagram of Fig. 13, showing jitter in the output data (measured at the actual I/O pins of the DRAM) while memory accesses are taking place. As can be seen, the error in data centering is only approximately 80 ps, while the peak-to-peak jitter is 250 ps, verifying the robustness of this architecture with respect to substrate and supply bounce.

The two DLL's shown in Fig. 14 together occupy 2400×250 μm^2 and draw 26 mA dc from a 2.5 V power supply.

Acknowledgment

The authors thank J. McBride, J. Cobrunson, T. Thrush, and B. Goble for their layout expertise. We also thank M. Horowitz, Y.-F. Chan, T. Sato, M. Uemura, K. Ohno, and S. Kamisaki for valuable help and stimulating discussions, and specially thank K. Tokushige.

References

[1] M. Horowitz *et al.*, "PLL design for a 500 MB/s interface," *IEEE ISSCC Dig.*, pp. 160–161, Feb. 1993.

[2] M. Johnson and E. Hudson, "A variable delay line PLL for CPU-coprocessor synchronization," *IEEE J. Solid-State Circ.*, pp. 1218–1223, Oct. 1988.

Thomas H. Lee received the S.B., S.M., and Sc.D. degrees in electrical engineering, all from the Massachusetts Institute of Technology, Cambridge, in 1983, 1985, and 1990, respectively, while serving as a consultant to a variety of companies.

He joined Analog Devices in Wilmington, MA, in 1990 where he was primarily engaged in the design of high-speed clock recovery devices. In 1992, he joined Rambus Inc., Mountain View, CA where he develops high-speed analog circuitry for 500 megabyte/s DRAM's. He is now an Assistant Professor of electrical engineering at Stanford University where his research interests are in low-power, high-speed analog circuits and systems. He continues an active relationship with Rambus Inc. as a consultant.

Kevin S. Donnelly was born in Los Angeles, CA in 1961. He received the B.S. degree in EECS from the University of California, Berkeley in 1985, and the M.S. degree in Electrical Engineering from San Jose State University in 1992.

Since 1984, he has worked at Memorex, Sipex, and National Semiconductor where he specialized in Bipolar and BiCMOS analog circuits for disk drive read/write and servo channels. In 1992, he joined Rambus Inc., Mountain View, CA, where he has designed high-speed CMOS PLL circuits for clock recovery and data synchronization. His interests include PLL's and DLL's, data converters, and audio circuits.

John T. C. Ho was born in Taipei, Taiwan, ROC in 1959. He received the B.S. degree in computer science from the Soochow University, Taipei, in 1982, and the M.S. degree in electrical engineering from University of Missouri, Columbia in 1988.

In 1988, he joined Altera Corporation, San Jose, CA where he worked on high-speed EPLD design. In 1990, he joined Tera MicroSystems Inc., Santa Clara, CA, as a circuit design engineer. In 1992, he joined Rambus Inc., Mountain View, CA, as a design engineer, where he has been involved in DLL design and Rambus ASIC interface design.

Jared Zerbe was born in New York, NY, in 1965. He received the B.S. degree in electrical engineering from Stanford University, Stanford, CA, in 1987.

He joined VLSI Technology Inc., in 1987, where he worked on custom and semicustom ASIC design until 1989. In 1989 he joined MIPS computer systems where was involved in the circuit design of CMOS microprocessors, primarily of high performance floating-point blocks. Since 1992 he has been with Rambus Inc., Mountain View, CA, and is presently engaged in designing high-speed logic, I/O, and clock-recovery circuits.

Mark G. Johnson was born in Houston, TX, in 1957. He received the BSEE degree from Rice University, Houston, in 1979, and the S.M. degree from Massachusetts Institute of Technology in 1982.

In 1982 he joined the memory design group at Mostek Corporation, where he designed a 1 Megabit CMOS dynamic RAM, and a family of high speed CMOS static RAM's. In 1986 he joined MIPS Computer Systems, Sunnyvale, CA, as a circuit designer. At MIPS he designed floating-point chips, phase locked loops, and high speed backplane buses. In 1992 he joined Rambus Inc., where he is presently engaged in the design of high speed I/O circuits, clock recovery, and DRAM interfaces.

Mr. Johnson is a member of Sigma Xi, Phi Beta Kappa, and Tau Beta Pi.

Toru Ishikawa was born in Yamagata Japan in 1961. He received the B.S.E.E. and M.S.E.E. degrees from Keio University, Tokyo, Japan in 1984 and 1986, respectively.

He joined the LSI memory division at NEC Corporation in 1986, where he designed a 1K channel time division switch, a BiCMOS static RAM, and a 4 Megabit CMOS dynamic RAM with low power refresh mode. He is presently engaged in designing DRAM cores for RDRAM devices.